САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

КАФЕДРА ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Плохов Артем Сергеевич

Выпускная квалификационная работа бакалавра

Использование нейросетевых методов для прогнозирования финансовых потоков

Направление 01.03.02

Прикладная математика и информатика

Научный руководитель,

ассистент

Малютин Е. А.

Санкт-Петербург

2018

Оглавление

Введение

В современном мире глобализация влияет на все сферы нашей жизни. События, произошедшие в одной точке мира, могут вызвать изменения на другом континенте, причем в некоторых случаях это может занять минуты или даже секунды.

Одной из областей, подверженных сильному влиянию глобализации, является фондовый рынок. Незначительные, на первый взгляд, происшествия могут решительным образом и в короткие сроки изменить цены на бирже, из-за чего перед трейдерами (торговцами на бирже) встаёт задача постоянно следить за состоянием рынка — малейшая оплошность может стоить целого состояния.

Однако человек не может бодрствовать круглосуточно, а усталость и недосып могут ухудшить мыслительные способности и навыки анализирования ситуации. Здесь на помощь человеку может прийти вычислительная техника — компьютеры прочно закрепились во всех областях нашей жизни, не нуждаются в сне и не страдают от усталости. Кроме того, по сравнению с человеком компьютер способен обрабатывать информацию гораздо быстрее, что в некоторых ситуациях может сыграть большое значение. В последнее время большую популярность получили искусственные нейронные сети, перенявшие у человека одну из важнейших способностей — способность к самообучению.

Первые искусственные нейронные сети были созданы ещё более пятидесяти лет назад, однако поначалу не нашли широкого применения. Но к концу двадцатого века началось возрождение интереса к искусственным нейронным сетям — вычислительная техника совершенствовалась, а сами сети имели важное преимущество по сравнению с классическими алгоритмами: они не требовали тщательного описания каждой ситуации и соответствующего ей действия, а могли обучаться на наборе данных,

подобно человеку. В результате появилось множество новых работ, описывающих применение нейронных сетей для решения самых разнообразных задач. Однако не все архитектуры одинаково полезны при решении различных задач — одна архитектура может успешно обучаться и применяться там, где другая будет совершенно бесполезна. При этом одна нейронная сеть, по-разному обученная, может порой справляться с задачами совершенно несхожими.

Интерес представляет возможность использования искусственных нейронных сетей в задачах прогнозирования изменения цен на фондовой бирже — по сравнению с человеком, нейронная сеть может быстрей реагировать на изменение ситуации, а также, при должном обучении, замечать зависимости, которые могут быть упущены человеком.

Целью данной работы является сравнение эффективности нейронных сетей с долгой краткосрочной памятью (LSTM) в задачах прогнозирования финансовых временных рядов по отношению к обычным рекуррентным нейронным сетям и сетям прямого распространения сигнала.

Постановка задачи

Целью работы является исследование эффективности использования искусственных нейронных сетей с долгой краткосрочной памятью в задачах прогнозирования финансовых временных рядов.

Для выполнения данной цели требуется выполнить следующие задачи:

1. Рассмотреть более ранние архитектуры искусственных нейронных сетей, такие как сети с прямым распространением сигнала и рекуррентные нейронные сети.
2. Изучить структуру и основные особенности сетей с долгой краткосрочной памятью, провести сравнение с более ранними архитектурами.
3. Построить сети с использованием упомянутых архитектур, произвести сравнение качества их работы в задачах прогнозирования финансовых временных рядов на примере валютной пары EUR-RUR.

Обзор литературы

Основной областью данной работы являются искусственные нейронные сети. В книге «Нейронные сети: полный курс» С. Хайкина [6] подробно описывается их строение, связь с биологическими нейронными сетями, основные архитектуры (в том числе — рекуррентные нейронные сети) и способы их обучения.

Особое внимание в работе уделено нейронным сетям с долгой краткосрочной памятью. Данный вид сетей был разработан с целью борьбы с проблемой угасающего градиента и описан в статье «Long short-term memory», Horichreiter S., Schmidhuber J. [4]

Подробный разбор принципа работы и взаимодействия ячеек сети LSTM, а также описание основных вариаций данной архитектуры были представлены в статье «Understanding LSTM Networks» [5].

Для построения нейронных сетей и оценки качества их работы была использована библиотека Keras [1], предоставляющая обширный набор средств проектирования нейронных сетей различных видов.

Глава 1. Общие сведения

1.1. Фундаментальный и технический анализ

На данный момент наиболее популярными методами прогнозирования цен на бирже являются фундаментальный и технический анализ.

Фундаментальный анализ строится на предположении, что цена на бирже не всегда отражает реальную стоимость акций – она может быть недооценена или переоценена – но стремится к ней. Следовательно, зная «действительную» цену акций, теоретически можно предсказать и будущую их стоимость на бирже.

Для определения «реальной» стоимости акций специалисты анализируют доступную в открытом доступе информацию о компании, стоимость акций которой требуется предсказать (например, сведения о заключаемых компанией сделках), новости о глобальных событиях, способных оказать влияние на развитие бизнеса, а также информацию, недоступную в открытых источниках (например, данные от личных источников внутри конкретной компании).

Трейдеры, использующие фундаментальный анализ, на основе полученной информации пытаются составить прогноз изменения цены в длительной перспективе. Таким образом, фундаментальный анализ более полезен при долгосрочных инвестициях, чем при спекуляциях на колебании цены.

Технический анализ для предсказания цены сравнивает текущую ситуацию на рынке с похожими ситуациями в прошлом и составляет прогноз, исходя из предположения, что цена будет изменяться так же, как изменялась в этих ситуациях.

В отличие от фундаментального анализа, технический анализ предполагает, что цена на бирже изменяется относительно быстро и практически сразу отображает действительную стоимость акции.

Наиболее распространённым инструментом в техническом анализе являются графики, учитывающие колебания цены и объём торгов и позволяющие выделить на них характерные повторяющиеся модели (паттерны). При совпадении текущей ситуации на бирже с одним из паттернов, можно предположить, что в и дальнейшем она будет вести себя похожим образом.

В отличие от фундаментального анализа, технический анализ чаще применяется для предсказания краткосрочных колебаний цены.

1.2. Искусственный нейрон

Искусственные нейронные сети, подобно биологическим нейронным сетям, состоят из *нейронов* – базовых единиц, связанных с другими такими же единицами, причём каждая связь – *синапс* – имеет соответствующий ей вес.

Строение нейрона представлено на *Рис. 1*:

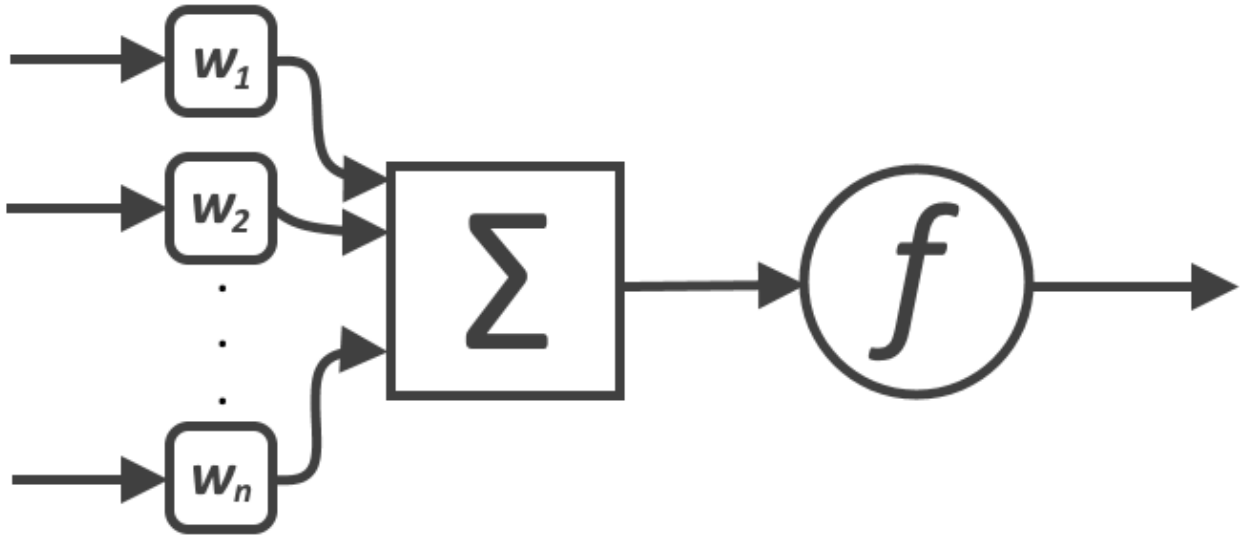


Рис. 1

Здесь w_1, w_2, ..., w_n – соответствующие связям веса, Σ – сумматор, f – функция активации.

Передача сигнала от одних нейронов к другим происходит следующим образом:

1. Нейрон принимает сигналы через входящие связи и умножает их на соответствующие связям веса.

2. Взвешенные сигналы суммируются, их сумма передаётся в качестве аргумента функции активации.

3. Значение, полученное с помощью функции активации, передаётся на выход нейрона и отправляется далее по сети.

Строение искусственной нейронной сети может меняться в зависимости от выполняемой сетью задачи, однако в большинстве сетей нейроны формируют *слои* – наборы нейронов одного уровня, использующих одинаковые функции активации и получающих сигналы от нейронов из других слоёв. Нейроны, принадлежащие одному слою, обычно не связаны.

Одной из важнейших особенностей искусственных нейронных сетей является способность к обучению. Обучение сети производится с помощью усиления или ослабления связей между нейронами путём изменения весов соответствующих связей. Наиболее распространённым методом обучения является *метод обратного распространения ошибки*, при котором на вход сети подаются значения из обучающей выборки, после чего оценка ошибки и коррекция в соответствии с ней весов выполняется, начиная с нейронов выходного слоя и продвигаясь последовательно от слоя к слою, вплоть до нейронов входного слоя (отсюда и название метода). Для применения данного метода обучения функции активации нейронов должны быть дифференцируемы.

В процессе обучения нейронной сети можно столкнуться с ситуацией, когда показатель ошибки на обучающей выборке продолжает уменьшаться, однако при применении сети на реальных данных ошибка начинает возрастать. Эта проблема называется *переобучением* сети и связана с тем, что сеть начинает чрезмерно подстраиваться под обучающую выборку. Такая ситуация может быть вызвана большим количеством эпох обучения или же излишним числом нейронов в самой сети.

Одним из способов борьбы с переобучением является метод под названием Dropout [3]. Суть метода заключается в игнорировании с некоторой вероятностью части нейронов в процессе обучения. Выбранные случайным образом нейроны никак не влияют на сеть на всех стадиях обучения. Делается это для того, чтобы разные нейроны не реагировали на одинаковые признаки. Так в процессе обучения создаётся своего рода комитет из нескольких сетей, возвращающий средний результат работы сетей.

Глава 2. Архитектуры нейронных сетей

В настоящее время существует множество архитектур нейронных сетей, нашедших применение в решении широкого спектра задач – в том числе в прогнозировании финансовых временных рядов. Рассмотрим некоторые из этих архитектур.

2.1. Многослойные сети прямого распространения

Многослойные сети прямого распространения состоят из входного, выходного и одного или нескольких скрытых слоёв. Сигнал принимается нейронами входного слоя и последовательно передаётся далее по сети от одного слоя к другому, пока не достигает выходного слоя.

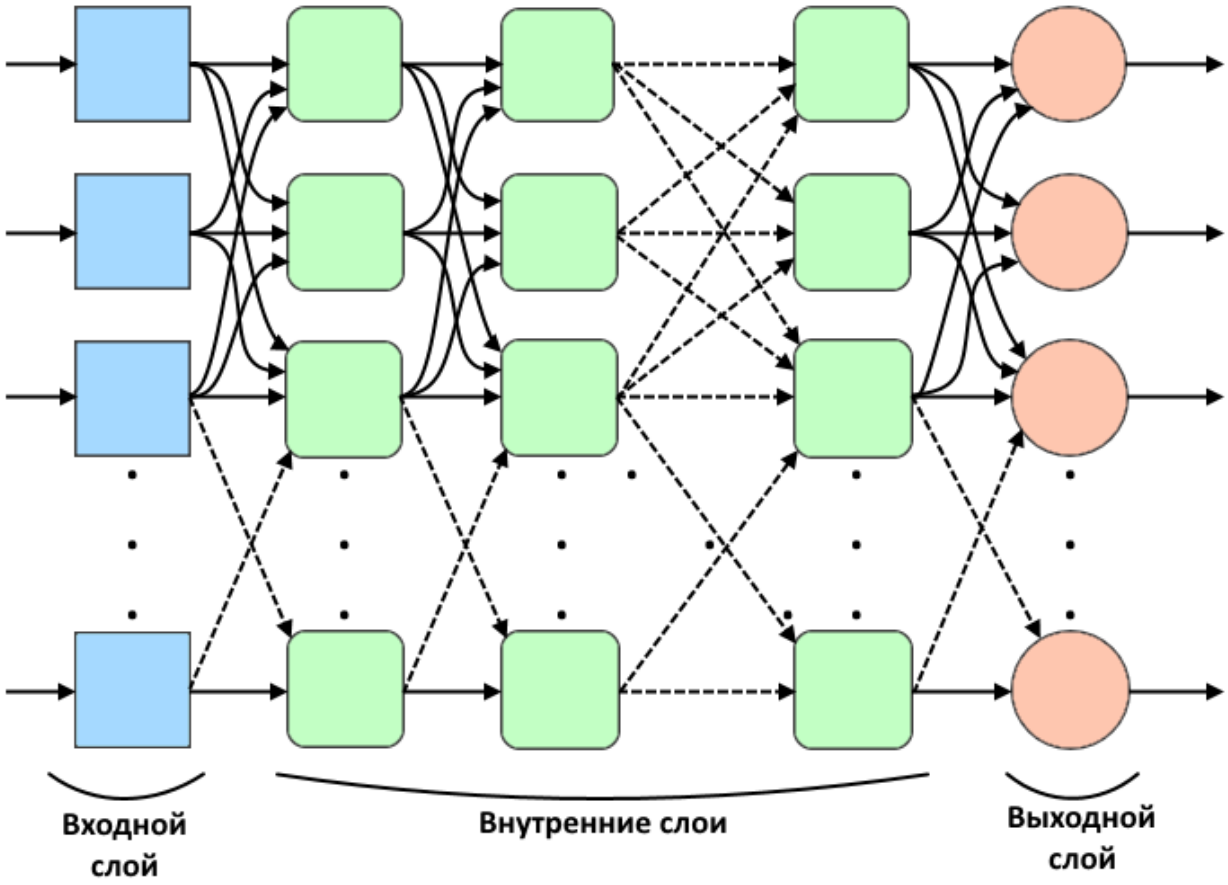


Рис. 2

Обычно нейроны в таких сетях связаны с предыдущим и следующим слоями по принципу «каждый с каждым», в то время как нейроны одного слоя друг с другом не связаны. Пример архитектуры такого типа представлен на *Рис. 2*.

2.2. Рекуррентные нейронные сети

Результат работы нейронных сетей прямого распространения основывается только на последних полученных входным слоем данных и не зависит от данных, которые были получены ранее. Каждый новый входной вектор такая сеть обрабатывает, как при первом запуске, независимо от сделанных ранее вычислений. Однако иногда может возникнуть необходимость учитывать предыдущие состояния сети – например, при обработке временных рядов или текста на естественном языке. В таких случаях полезно использовать нейронную сеть, которая могла бы сохранять информацию о её предыдущих состояниях и использовать эту информацию на последующих этапах работы.

Рекуррентной нейронной сетью (Recurrent Neural Network, RNN) называется сеть, имеющая как минимум одну обратную связь. В такой сети нейрон может передавать информацию на вход предыдущим нейронам или самому себе, влияя, таким образом, на итоговый результат работы всей сети.

Модель простейшей рекуррентной нейронной сети представлена на *Рис. 3*. Нейрон скрытого слоя передаёт на свой вход информацию о своём состоянии в конце предыдущей итерации [2]. В более сложных архитектурах эта информация можете передаваться также соседним нейронам.

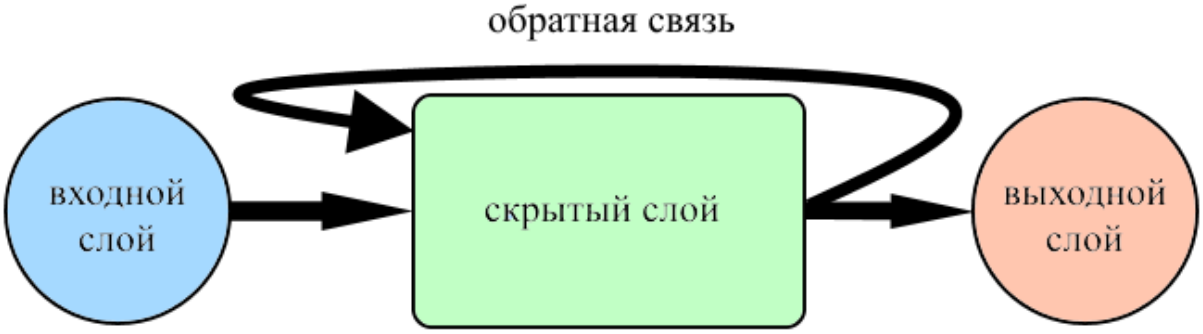


Рис. 3

Такая архитектура может быть условно представлена в виде сети прямого распространения, в каждом слое содержащей n нейронов, где n – количество итераций с начала работы сети (*Рис. 4*).

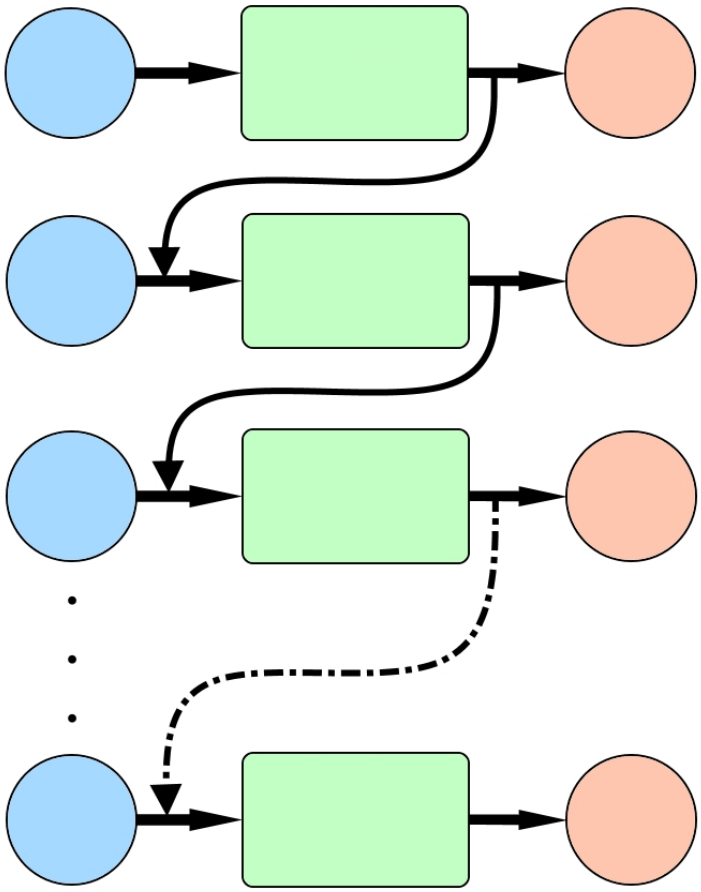


Рис. 4

2.3. Сети с долгой краткосрочной памятью

При работе с рекуррентными нейронными сетями (и глубокими нейронными сетями вообще) можно столкнуться с ситуацией, когда градиент начинает стремительно угасать (или же, наоборот, возрастать). Данная проблема называется *проблемой угасающего (взрывного) градиента* или *проблемой обращения градиентов в ноль*. В этом случае становится проблематично обучить нейронную сеть предсказывать результат на основе входных данных, относящихся к далёкому для предсказываемого момента прошлому, в результате сеть быстро «забывает» информацию о предыдущих состояниях, учитывая лишь недавние. Это может стать преградой при решении задач, в которых требуется принимать во внимание информацию, полученную сравнительно давно – например, при обработке текста на естественном языке контекст может зависеть от сказанного в предыдущем предложении или даже предыдущем абзаце, однако из-за потери информации о предшествующих состояниях нейронная сеть будет учитывать лишь слова последнего предложения, теряя общий смысл.

Сети с долгой краткосрочной памятью (Long Short-Term Memory, LSTM), предложенные З. Хохрайтером и Ю. Шмидхубером [4], были созданы для решения проблемы угасающего градиента. Роль нейронов в такой сети играют ячейки, сами по себе являющиеся небольшими нейронными сетями. Для сохранения ценных данных на протяжении длительного времени эти ячейки используют специальные фильтры, называющиеся *вентилями* или *шлюзами* (gates). Таких вентилей три: входной, выходной и вентиль забывания. Входной вентиль при получении новой информации определяет, стоит ли нейрону её запоминать и если да, то какую её часть. Выходной вентиль позволяет определить, какая часть из находящейся в памяти информации важна в данный момент, а какой следует пренебречь. И, наконец, вентиль забывания помогает сети удалить из памяти

ту информацию, которая уже не пригодится в дальнейшей работе [5]. Строение ячейки такой сети представлено на *Рис. 5*.

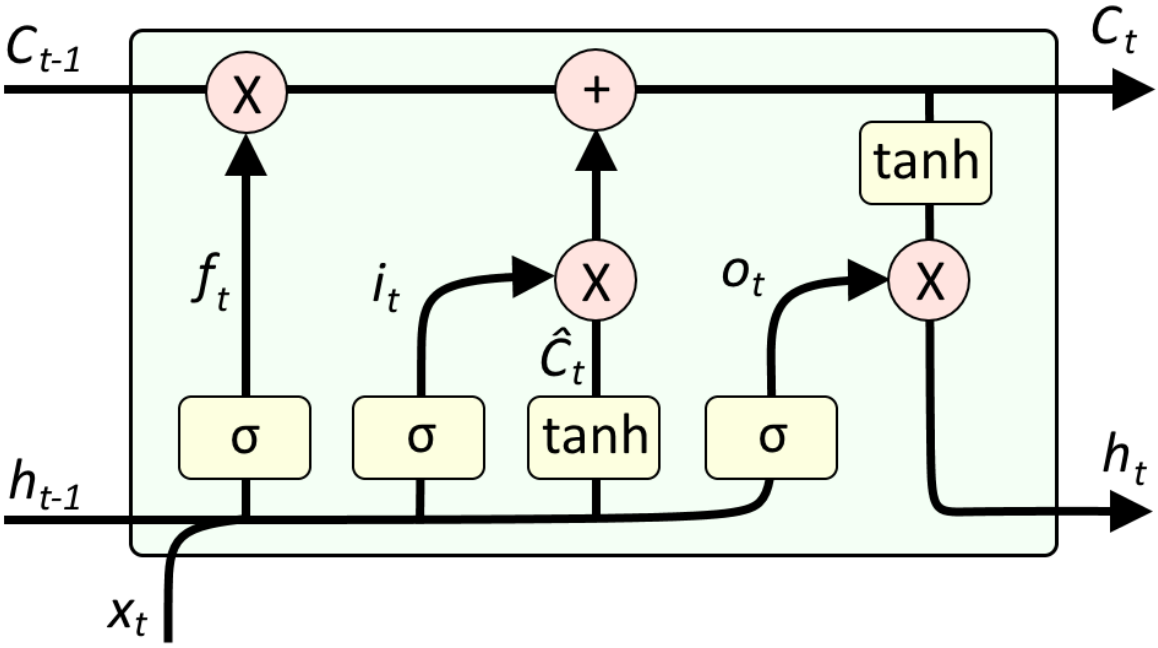


Рис. 5

Сначала ячейке требуется найти данные, которые перестали быть актуальными и от которых требуется избавиться. Выходные данные предыдущего шага и входные данные текущего шага передаются на слой вентиля забывания с сигмоидальной функцией активации f_t, которая возвращает число от 0 до 1 для всех значений предыдущего состояния C_t. Таким образом определяется, какие данные останутся в памяти, а какие будут забыты. При получении в качестве значения нуля данные полностью стираются, при получении единицы – полностью сохраняются.

$$W_f \cdot [h_{t-1}, x_t] + b_f,$$
$$f_t = \sigma$$

где W_f – вектор весов, b_f – смещение.

Затем нужно определить, какие данные будут добавлены в ячейку. Данная задача выполняется в два этапа: на первом этапе слой входного

вентиля с сигмоидальной функцией активации i_t решает, какие из уже имеющихся в памяти данных нужно заменить; на втором этапе слой с гиперболическим тангенсом в качестве функции активации $\hat{C}_t$ составляет набор значений, которые могут быть добавлены в память ячейки, после чего информация в памяти ячейки заменяется на новую.

$$\hat{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

Определив, какие данные заслуживают остаться в памяти, а какие – нет, мы должны обновить состояние ячейки. Для этого умножаем предыдущее состояние на результат работы слоя вентиля забывания f_t, избавляясь, таким образом, от устаревшей информации. После этого добавляем новую информацию – прибавляем к имеющимся значениям $i_t \cdot \hat{C}_t$ (кандидаты на добавление, полученные при помощи слоя входного вентиля).

Обновив своё состояние, ячейка готова передавать данные на выход, однако сначала нужно определить, какая именно информация должна быть переданы. Для этого входные данные подаются на слой с сигмоидальной функцией активации o_t, а состояние ячейки – на слой с активацией в виде гиперболического тангенса h_t, выдающего значения в диапазоне от -1 до 1.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \cdot \tanh(C_t)$$

В качестве выходного значения ячейка возвращает произведение результатов работы этих слоёв.

Сети с долгой краткосрочной памятью нашли применение в обработке текстов, написанных на естественном языке, задачах распознавания речи и прогнозирования временных рядов.

2.4. Функции активации

Линейная функция возвращает аргумент, умноженный на параметр t:

$$f(x) = tx$$

Линейная функция с насыщением:

$$f(x) = \begin{cases} 0, & \text{если } x \leq 0 \\ 1, & \text{если } x \geq 1 \\ x, & \text{иначе} \end{cases}$$

Линейные функции используются, в основном, в качестве функций активации нейронов входного слоя.

Пороговая функция или функция Хевисайда обращается в нуль, если аргумент меньше заданного порогового значения T, и принимает значение единицы, когда аргумент превышает или равен этому значению:

$$\begin{cases} 1, & \text{если } x \geq T \\ 0, & \text{иначе} \end{cases}$$

Так как функция принимает лишь два значения, её применение обычно ограничено задачами классификации. Не является дифференцируемой на всей числовой оси, поэтому не может быть использована в сетях, обучаемых с помощью алгоритма обратного распространения ошибки.

Сигмоида – один из наиболее часто используемых типов функций активации. Используется, в основном, в качестве функции активации нейронов внутренних слоёв. В отличие от пороговой функции, может принимать не только значения «1» и «0», но и промежуточные, что расширяет область задач, в которых она может быть использована.

Распространённые сигмоиды:

Логистическая функция, принимает значения в интервале (0, 1):

$$\sigma(x) = \frac{1}{1 + e^{-tx}}$$

где параметр t определяет крутизну функции. При стремлении t к бесконечности функция вырождается в пороговую функцию, при $t = 0$ – в константу.

Гиперболический тангенс, в отличие от логистической функции, принимает значения из интервала (-1; 1):

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

По сравнению с прочими описанными функциями активации, обучение сетей с гиперболическим тангенсом в качестве функции активации является наиболее затратным в плане вычислительной мощности.

ReLU (rectified linear unit) равна нулю для всех аргументов, меньших или равных нулю, и совпадает с линейной функцией для аргументов больше нуля:

$$f(x) = max(0, x)$$

Leaky ReLU для отрицательных значений аргумента возвращает аргумент, умноженный на малый параметр a (например, $a = 0.01$):

$$f(x) = \begin{cases} x, & \text{если } x > 0 \\ ax, & \text{иначе} \end{cases}$$

2.5. Функции ошибки

Функция ошибки позволяет оценить, насколько предсказанный нейронной сетью результат отличается от ожидаемого. В зависимости от выбора функции ошибки может изменяться оценка правильности предсказаний сети во время обучения и, следовательно, результат всего обучения. Поэтому для разных задач могут требоваться различные функции ошибки.

Введём следующие обозначения:

z_i – ожидаемый результат работы сети на шаге i

y_i – полученный результат работы сети на шаге i

n – общее количество шагов

Среднеквадратическая ошибка (MSE)

$$\frac{\sum_{i=1}^{n}(z_i - y_i)^2}{n}$$

Средняя абсолютная ошибка (MAE)

$$\frac{\sum_{i=1}^{n}|z_i - y_i|}{n}$$

Глава 3. Экспериментальная часть

3.1. Рабочая среда

Для проведения экспериментов использовался персональный компьютер со следующими характеристиками:

1. Процессор Intel Core 2 Duo E7300 – 2.66GHz (x2)
2. Видеокарта NVIDIA GeForce GT 520
3. Операционная система Linux Mint 18.1

Программная часть выполнена на языке Python 3, для работы с данными были использованы следующие библиотеки:

- NumPy – библиотека для создания и работы с массивами данных
- Pandas – библиотека для работы с числовыми таблицами, работает поверх библиотеки NumPy
- scikit-learn – инструменты для решения задач классификации, кластеризации и регрессии, а также начальной обработки данных
- Matplotlib – библиотека для построения графиков
- Keras с использованием Theano в качестве бэкенда

Keras – открытая библиотека, написанная на языке Python и нацеленная на создание нейронных сетей и упрощение работы с ними. Может работать с библиотеками Theano и TensorFlow, позволяя при возникновении надобности переключаться между ними, не меняя при этом код программы.

При проведении экспериментов Keras запускался вместе с библиотекой Theano, так как она показала более высокую производительность на рабочей машине.

Keras и Theano позволяют использовать при работе вычислительные мощности графических процессоров. Однако по причине слабых

характеристик и малой вычислительной мощности установленной в рабочую машину видеокарты данная функция не была использована и все вычисления производились при помощи центрального процессора.

3.2. Обработка данных

Для обучения и тестирования нейронных сетей были загружены данные о котировках валютной пары EUR-RUR с 01.01.2013 по 31.12.2017, доступные в открытом доступе на сайте finam.ru.

Загруженные данные имели следующие поля:

Дата, время, цена открытия, максимальная цена, минимальная цена, цена закрытия, объём торгов.

Пример загруженных данных на *Рис. 6*:

```
20170102,000000,64.3580000,64.7322000,64.3490000,64.3610000,126
20170102,010000,64.3550000,64.7418000,64.3430000,64.7335000,95
20170102,020000,64.3670000,64.8818000,64.3430000,64.7369000,131
20170102,030000,64.7393000,64.8690000,64.3930000,64.7295000,88
20170102,040000,64.7210000,64.7347000,64.6050000,64.6360000,79
20170102,050000,64.6330000,64.6973000,64.6250000,64.6776000,100
20170102,060000,64.6240000,64.6785000,64.5930000,64.6585000,110
20170102,070000,64.6090000,64.6816000,64.3920000,64.6727000,132
20170102,080000,64.6060000,64.7512000,64.5840000,64.6441000,112
20170102,090000,64.5830000,64.6820000,64.5682000,64.5720000,95
20170102,100000,64.6720000,64.6760000,64.4220000,64.5893000,125
20170102,110000,64.6750000,64.7020000,64.3364000,64.3637000,132
20170102,120000,64.4230000,64.4580000,64.1901000,64.1901000,127
20170102,130000,64.3170000,64.6160000,64.0110000,64.2250000,156
20170102,140000,64.6080000,64.6100000,64.1732000,64.2387000,126
20170102,150000,64.2970000,64.3040000,64.0023000,64.1415000,135
20170102,160000,64.1700000,64.2380000,64.0284000,64.0284000,135
20170102,170000,64.1030000,64.1270000,63.9807000,64.0589000,131
20170102,180000,64.0950000,64.1750000,64.0503000,64.1008000,132
20170102,190000,64.1130000,64.1600000,64.0463000,64.0840000,136
20170102,200000,64.1530000,64.1930000,64.0680000,64.0759000,120
20170102,210000,64.0790000,64.1400000,64.0315000,64.0520000,110
20170102,220000,64.0720000,64.1260000,64.0110000,64.0236000,119
20170102,230000,64.0920000,64.1090000,64.0110000,64.0122000,102
```

Рис. 6

Так как нейронная сеть плохо воспринимает ненормированные данные, была произведена предварительная обработка данных. Поля даты и времени

были отброшены, поскольку предоставленные данные имели фиксированный интервал между записями, равный одному часу. Затем была произведена стандартизация данных всех оставшихся полей, в результате данные были преобразованы в значения, лежащие на отрезке [-1; 1].

3.3. Использованные архитектуры

Для проведения экспериментов на базе библиотеки Keras были построены искусственные нейронные сети трёх видов:

- Нейронная сеть с прямым распространением сигнала
- Рекуррентная нейронная сеть
- Сеть с долгой краткосрочной памятью

Для каждого вида было создано несколько отдельных сетей, отличающихся друг от друга количеством внутренних слоёв и количеством нейронов в слоях.

Также для борьбы с переобучением в ряде случаев был использован метод Dropout.

В качестве функции ошибки использовалась среднеквадратичная ошибка, для обучения применялся оптимизатор Adam. В качестве функции активации сетей с прямым распространением использовалась ReLU, для рекуррентных сетей и сетей LSTM – гиперболический тангенс.

Программное описание структуры сетей представлено в Приложение 1.

3.4. Результаты экспериментов

Выборка малого размера

При использовании выборки размера 100 и обучении сетей на протяжении 60 эпох были получены схожие результаты для сетей всех видов, различия в точности прогнозирования незначительны (Приложение 2).

Однако скорость обучения сетей значительно отличается: среднее время обучения одной сети LSTM составило 30 секунд, в то время как обучение рекуррентной сети производилось за 11 с., сети с прямым распространением – за 2 с.

Среднеквадратичная ошибка сетей на выборке малого размера представлена в Таблица 1:

LSTM	LSTM + Dropout	RNN	RNN + Dropout	FFNN	FFNN + Dropout
0.0649572	0.0655432	0.0860737	0.0879772	0.0726678	0.0584342

Таблица 1

Выборка среднего размера

При использовании выборки размера 500 наибольшей точности добились рекуррентные нейронные сети (Приложение 3). Особенно хорошо это заметно при использовании сетей, использующих слои из 120 нейронов и обученных за 30 эпох. В этом случае хорошие результаты были показаны сетями как с Dropout, так и без него.

При этом сети LSTM показали худший результат как в плане точности, так и по времени обучения.

Среднеквадратичная ошибка и время обучения сетей на выборке среднего размера представлены в Таблица 2:

	LSTM	LSTM + Dropout	RNN	RNN + Dropout	FFNN	FFNN + Dropout
Нейроны: 40, эпохи: 60	0.4173333 20 с.	0.4729796 21 с.	0.0314153 10 с.	0.1348052 12 с.	0.0686822 2 с.	0.0777155 2 с.
Нейроны: 80, эпохи: 60	0.2526756 26 с.	0.3964917 30 с.	0.0367121 14 с.	0.1594309 18 с.	0.1308449 2 с.	0.2312115 3 с.
Нейроны: 120, эпохи: 30	0.2474819 21 с.	0.2862170 22 с.	0.0233408 9 с.	0.0473693 9 с.	0.0920971 1 с.	0.1232884 1 с.
Нейроны: 120, эпохи: 90	0.2575455 42 с.	0.3754761 43 с.	0.1013059 13 с.	0.1844331 15 с.	0.1068751 3 с.	0.2346533 3 с.

Таблица 2

Выборка большого размера

На выборке размера 1000 большинство сетей стали проявлять признаки переобучения. Особенно сильно это было выражено для сетей LSTM, не использующих Dropout. Однако сети LSTM с методом Dropout, напротив, показали более высокую точность и проявили признаки переобучения лишь при увеличении количества нейронов в слое до 150 (Приложение 4).

Как и в предыдущих случаях, сети LSTM требовали наибольшего времени обучения.

Среднеквадратичная ошибка и время обучения сетей на выборке большого размера представлены в Таблица 3:

	LSTM	LSTM + Dropout	RNN	RNN + Dropout	FFNN	FFNN + Dropout
Нейроны: 40	1.6265556 117 с.	0.0608687 143 с.	0.2176605 69 с.	0.4253625 68 с.	0.5198941 19 с.	0.3040569 24 с.
Нейроны: 80	3.2279576 156 с.	0.0221036 163 с.	0.3911052 61 с.	0.6051075 63 с.	0.6190428 19 с.	0.8494295 24 с.
Нейроны: 150	2.0411882 401 с.	0.1398953 406 с.	0.3614659 71 с.	0.8879787 81 с.	0.7346955 20 с.	0.3750998 25 с.

Таблица 3

Выводы

По результатам экспериментов можно установить следующее:

- Сети с прямым распространением сигнала показали наименьшую точность прогнозирования, но были наименее требовательны к вычислительным ресурсам в процессе обучения.

- Рекуррентные нейронные сети показали лучший результат при работе с небольшими обучающими выборками и наибольшую точность прогнозирования в целом; при использовании объёмных обучающих выборок было заметно влияние переобучения. Данный вид сетей занимает промежуточное положение по требовательности к вычислительным ресурсам во время обучения.

- Нейронные сети с долгой кратковременной памятью показали лучший результат при обучении на выборках большого размера и использовании метода Dropout, значительно сокращающего влияние переобучения при использовании объёмных выборок, однако уступили рекуррентным нейронным сетям по показателю минимальной ошибки. Кроме того, они показали худшее время обучения. Это связано с более сложной структурой и использованием в качестве функции активации гиперболического тангенса.

Заключение

В данной работе были рассмотрены особенности искусственных нейронных сетей с долгой краткосрочной памятью и проведено сравнение сетей этой архитектуры с нейронными сетями прямого распространения сигнала и рекуррентными нейронными сетями.

Результаты сравнения показали, что при использовании метода Dropout нейронные сети с долгой краткосрочной памятью имеют более высокую устойчивость к переобучению и показывают хорошие результаты на выборках большого размера, однако не смогли показать значительного улучшения общей точности прогнозирования по сравнению с простыми рекуррентными сетями. Кроме того, в силу более сложной архитектуры данные сети требуют значительно большего времени на их обучение.

Данная архитектура и её способность учитывать факторы из далёкого прошлого может оказаться полезной при использовании фундаментального анализа. В качестве же инструмента технического анализа, анализирующего сравнительно близкие по времени параметры, более предпочтительным вариантом является использование простых рекуррентных сетей, предоставляющих схожую точность при меньших требованиях к ресурсам.

Список литературы

1. About Keras models. https://keras.io/models/about-keras-models/
2. Elman J. L. – Finding Structure in Time // Cognitive Science, 1990. 14, 179 – 211
3. Hinton G. E., Srivastava N., Krizhevsky A., Sutskever I., Salakhutdinov R. R. – Improving neural networks by preventing co-adaptation of feature detectors // arXiv: 1207.0580v1, 2012
4. Horichreiter S., Schmidhuber J. – Long short-term memory // Neural Computation, 1997. 9(8): 1735 – 1780
5. Understanding LSTM Networks. http://colah.github.io/posts/2015-08-Understanding-LSTMs/
6. Хайкин С. Нейронные сети: полный курс, 2-е издание. М.: «Вильямс», 2006, 1104 с.

Приложение

Приложение 1

Программное описание рассмотренных моделей

```
      def md(tp, units, shape, activation = 'tanh', r_activation =
'hard_sigmoid', loss_func = 'mean_squared_error', optimizer = 'adam'):
            model = Sequential()
            if tp == 'LSTM':
                  model.add(LSTM(units, activation = activation,
recurrent_activation = r_activation, input_shape = shape))
                  model.add(Dense(1, activation = 'linear'))
            elif tp == 'LSTM_drop':
                  model.add(LSTM(units, activation = activation,
recurrent_activation = r_activation, input_shape = shape))
                  model.add(Dropout(0.3))
                  model.add(Dense(1, activation = 'linear'))
            elif tp == 'FFNN':
                  model.add(Dense(units, activation = 'relu', input_dim =
shape))
                  model.add(Dense(1, activation = 'linear'))
            elif tp == 'FFNN_drop':
                  model.add(Dense(units, activation = 'relu', input_dim =
shape))
                  model.add(Dropout(0.3))
                  model.add(Dense(1, activation = 'linear'))
            elif tp == 'RNN':
                  model.add(SimpleRNN(units, activation = activation,
input_shape = shape))
                  model.add(Dense(1, activation = 'linear'))
            elif tp == 'RNN_drop':
                  model.add(SimpleRNN(units, activation = activation,
input_shape = shape))
                  model.add(Dropout(0.3))
                  model.add(Dense(1, activation = 'linear'))


            model.compile(loss = loss_func, optimizer = optimizer)
            return model
```

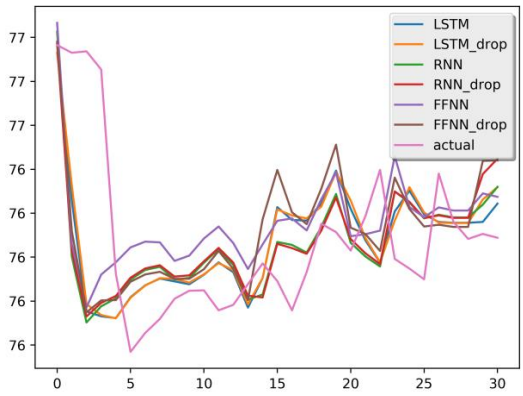
Приложение 2

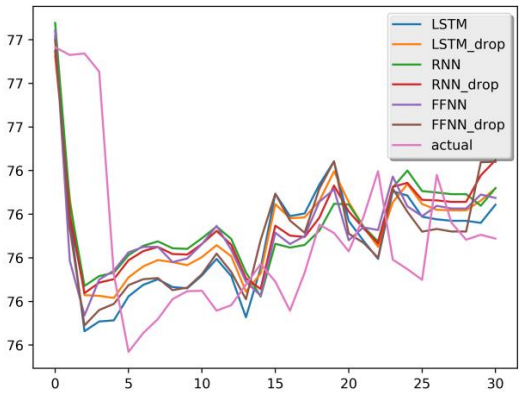
Сравнение работы сетей на малой выборке.

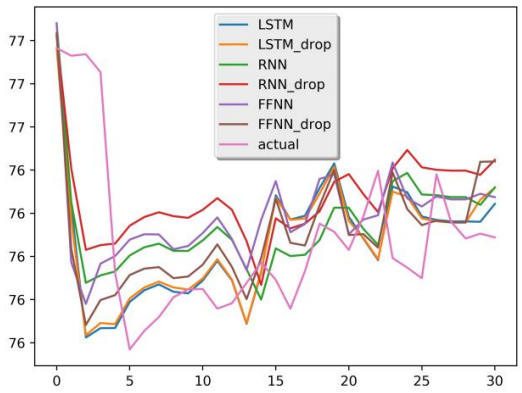
Размер выборки: 100, количество эпох обучения: 60.

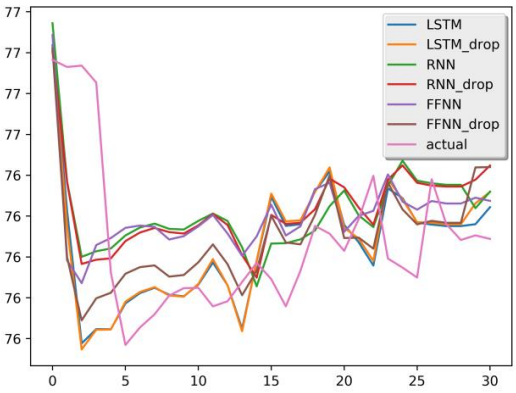
На графиках ниже представлены результаты работы обученных сетей, в порядке увеличения количества нейронов в слое (слева направо, сверху вниз): 40, 80, 120, 150 нейронов.

Различия в точности результатов незначительны.



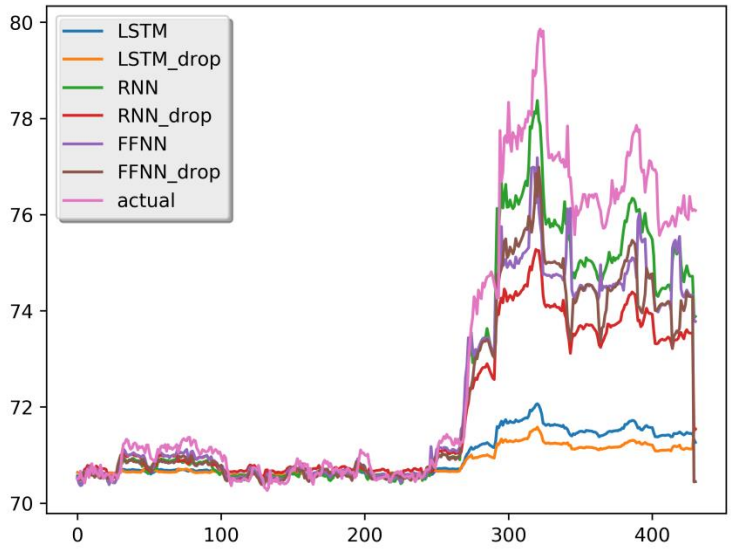






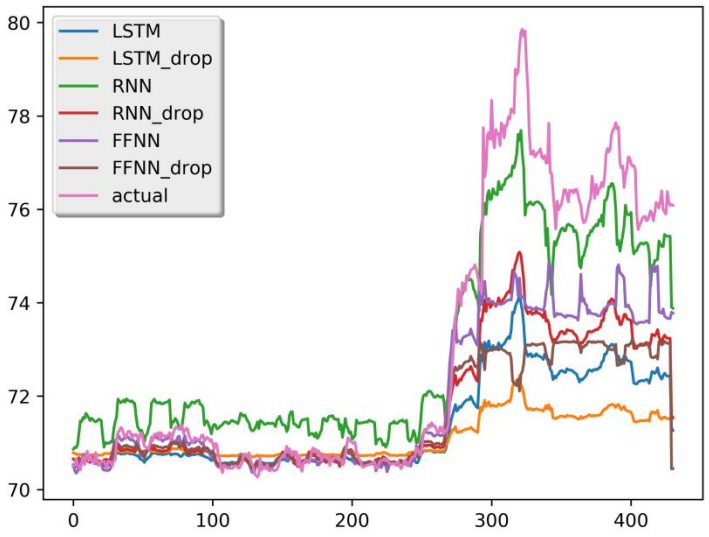
Приложение 3

Сравнение работы сетей на выборке среднего размера.



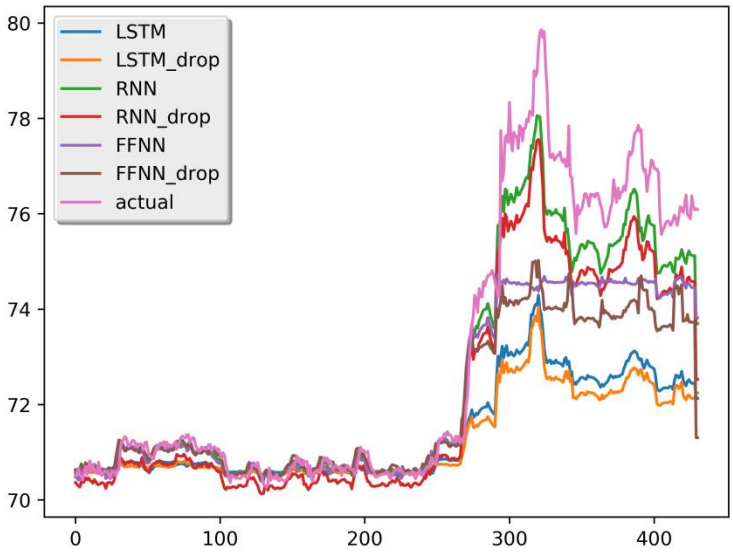
Размер выборки: 500, нейронов в слое: 40, эпох: 60.

Наибольшая точность у рекуррентной сети, наименьшая – у LSTM.



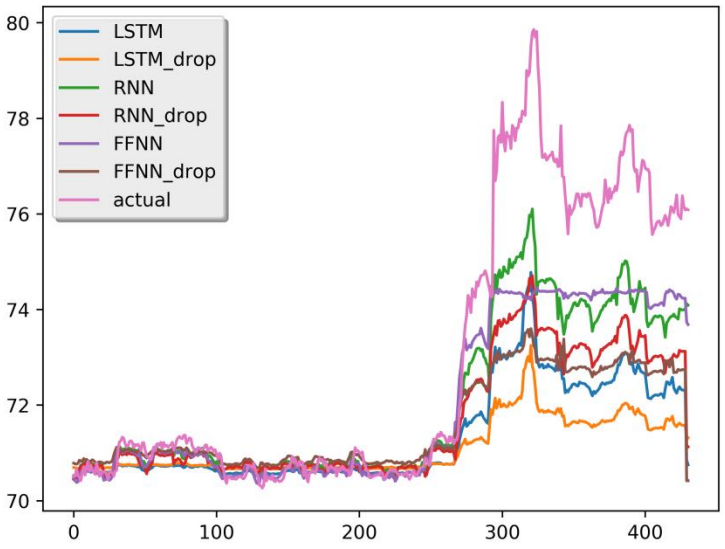
Размер выборки: 500, нейронов в слое: 80, эпох: 60.

При увеличении числа нейронов всех сетей, кроме LSTM, наблюдается переобучение и понижение точности; у LSTM – повышение точности.



Размер выборки: 500, нейронов в слое: 120, эпох: 30.

Высокая точность у обеих рекуррентных сетей, повышение точности LSTM с Dropout.

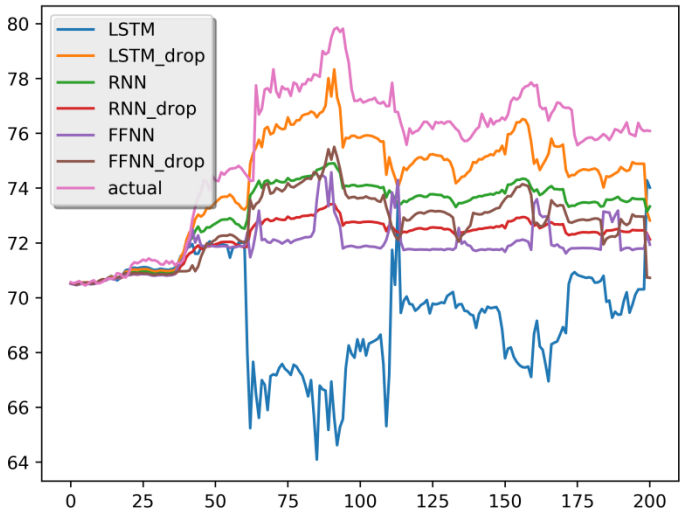


Размер выборки: 500, нейронов в слое: 120, эпох: 90.

Наблюдается переобучение сетей, снижение точности.

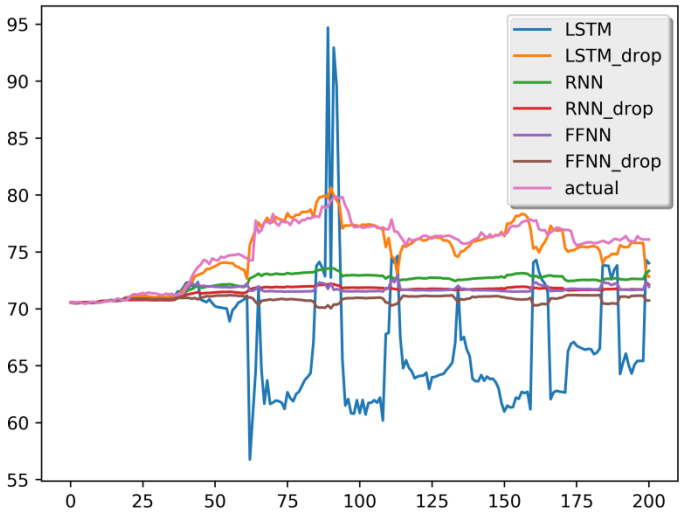
Приложение 4

Сравнение работы сетей, обученных на выборке большого размера.



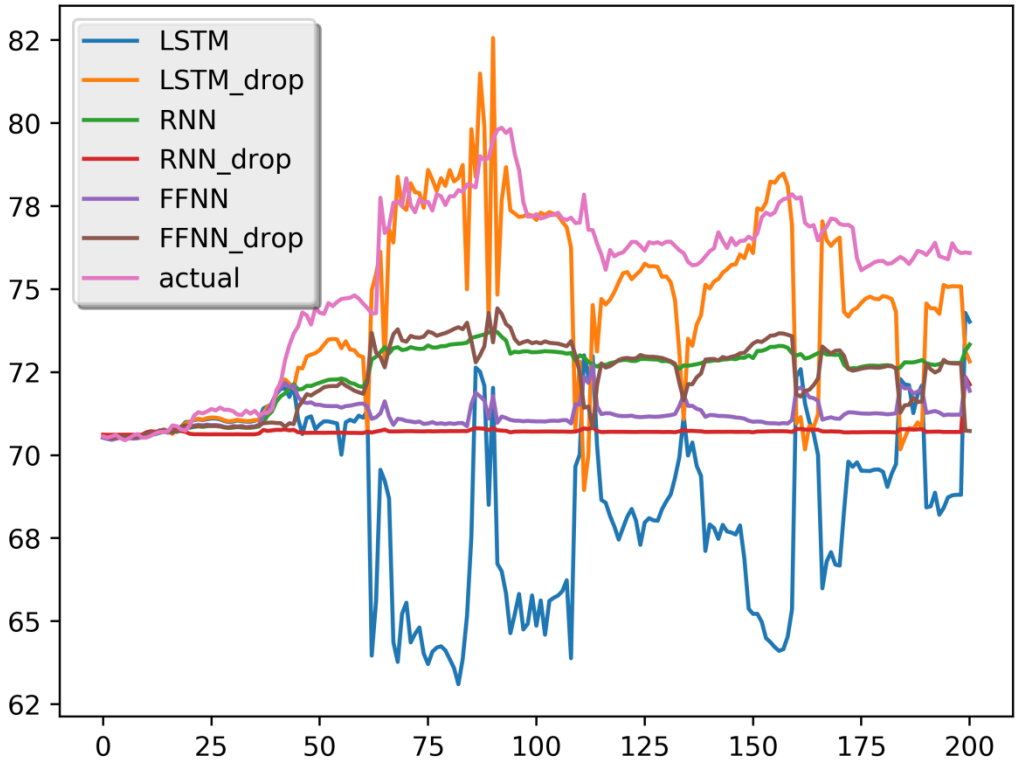
Размер выборки: 1000, нейронов в слое: 40, эпох: 60.

Наблюдается переобучение сетей, за исключением LSTM с Dropout, показывающей наилучшую точность.



Размер выборки: 1000, нейронов в слое: 80, эпох: 60.

Дальнейшее ухудшение точности всех сетей, кроме LSTM с Dropout; улучшение точности LSTM с Dropout.



Размер выборки: 1000, нейронов в слое: 150, эпох: 60.

Переобучение и сильное снижение точности всех сетей, включая LSTM с Dropout.